Adding a filter for the PU21 transform

FFmpeg GSoC 2023 Proposal

Mentor : **Thilo Borgmann**

Applicant : Rajiv Harlalka

Table of Contents

Contact Information

Name	:	Rajiv Harlalka
Email	:	rajivharlalka009@gmail.com
IRC	:	@rajivharlalka:Libera.IRC
GitHub	:	rajivharlalka
Phone	:	+91 8987966604
Website:		https://rajivharlalka.in
Twitter	:	rajivharlalka09
Address	:	Azad Hall, Indian Institute of Technology Kharagpur - 721302 West Bengal, India
Location	:	Kolkata, India (GMT +5:30)
Age	:	22

Personal Information

I’m Rajiv Harlalka, a fourth-year undergraduate student at the Indian Institute of Technology, Kharagpur (India). I am a member of Kharagpur Open Source Society, a university club which organizes lectures and events for the students of IIT Kharagpur about different aspects of Software Engineering and working in a Community. Open-source projects have made me realize the importance of documented and easy-to-understand code.

I worked as a student developer in Kharagpur Winter of Code conducted by Kharagpur Open Source Society, IIT Kharagpur. It is an event in December where students can work on projects of their interest.

I was also a student of the Google Summer of Code 2023 batch under the PostgreSQL organization wherein I helped in extending functionalities to a PostgreSQL extension for collecting and visualizing statistics. I had great learning during this time and got to meet a number of people from the community. I even got a chance to speak on the development of the project at PostgreSQL’s annual conference PG Conf 2023 held in Prague in the month of December 2023.

I have firm C, C++, Golang and Javascript knowledge. I have been fascinated by the famous Go Proverb “Clear is better than clever,” which I admire, especially in open source code base. Precise, reusable functions, so readable that documentation isn’t needed often, impress me greatly. I use Arch Linux as my daily operating system, with VScode and NeoVim being my most used text editors. I try being a Terminal Ninja, trying to do things which could be done in seconds using an alternative GUI tool. Getting those flags correct with man pages brings me a sense of pleasure and satisfaction.

Project Description

Title	:	Adding a filter for the PU21 transform
Duration	**:**	350 hours
Time	**:**	12 weeks
Technologies	**:**	C, Make, Linux
Topics	**:**	Image Processing, FFmpeg Filters

Abstract

Video encoding-decoding standards continue evolving, aiming for higher compression efficiency and better visual and audible qualities. FFmpeg project has played a pivotal role in shaping the multimedia industry. Filters lie at the core of how FFmpeg can perform manipulations on a multimedia file, whether finding information out of the media or transforming it into a required one.

A new encoding function (PU21) has been developed to convert absolute high dynamic range (HDR) linear color values into approximately perceptually uniform (PU) values, which can be used with standard quality metrics. This transformation would eventually help evaluate HDR videos based on quality tests known for SDR ones.

Motivation

Perpetual uniform encoding is an approach that encodes high dynamic range (HDR) images and videos. The encoding converts absolute linear RGB color values into approximately perceptually uniform values. The conversion means they more closely match how human vision perceives color brightness and contrast. With the help of this encoding, metrics such as PSNR and SSIM have become helpful for HDR media, too, and have been a standard for the SDR ones. The recent PU21 encoding considers contrast sensitivity, coding artifacts, and glare effects, improving the accuracy of quality predictions for standard metrics when correlated with subjective scores on HDR images.

Proposed Deliverables (during GSoC)

1. Addition of a new filter (PU21 transform) to the filters library
2. Addition of tests for the new filter
3. Fortnightly blogs on developmental advances and milestones

Project Breakdown

This topic explains the work proposed under each deliverable:

Addition of a new filter (PU21 transform) to the filters library

PU21 transform is an advancement of the previously devised PU08 transform by extending contrast sensitivity, modeling realistic coding artifacts, and accounting for glare. Rafał K. Mantiuk et al. paper PU21: A novel perceptually uniform encoding for adapting existing quality metrics for HDR references some quantified approaches of handling different problems, such as banding and glare, and proposes some metrics and functions on leveraging them. This would be useful for generating the actual logic for the PU21 transform. I would build upon the original stub created and extend it to add functionalities of the actual PU21 transform.

No-Operation Stub

```
#include <libavfilter/avfilter.h>
#include <libavutil/opt.h>
#include <libavutil/pixdesc.h>
#include <libavutil/imgutils.h>
#include <libavfilter/formats.h>
#include <libavfilter/internal.h>

typedef struct PU21Context {
  const AVClass* class;
  double L_min, L_max;
  double par[7];
  char* type;
} PU21Context;

#define OFFSET(x) offsetof(PU21Context, x)
```

```
#define FLAGS AV_OPT_FLAG_FILTERING_PARAM|AV_OPT_FLAG_VIDEO_PARAM
static const AVOption pu21_options[] = {
    { "type", "Set the type of encoding", OFFSET(type), AV_OPT_TYPE_STRING, {.str =
"banding_glare"}, 0, 0, FLAGS }, // options can be banding, banding_glare, peaks,
peaks_glare
    { NULL }
};

AVFILTER_DEFINE_CLASS(pu21);

static av_cold int pu21_init(AVFilterContext* ctx) {
  PU21Context* pu21 = ctx->priv;
  pu21->L_min = 0.005;
  pu21->L_max = 10000;

  // These are the default parameters for the banding_glare encoding, would add rest of
the parameters using switch case based on pu21->type
  pu21->par[0] = 0.353487901;
  pu21->par[1] = 0.3734658629;
  pu21->par[2] = 8.277049286 * pow(10, -5);
  pu21->par[3] = 0.9062562627;
  pu21->par[4] = 0.09150303166;
  pu21->par[5] = 0.9099517204;
  pu21->par[6] = 596.3148142;
  return 0;
}

static int filter_frame(AVFilterLink* inlink, AVFrame* frame) {
  AVFilterContext* ctx = inlink->dst;
  PU21Context* pu21 = ctx->priv;

  int x, y;
  for (y = 0; y < frame->height; y++) {
    for (x = 0; x < frame->width; x++) {
      // actual pu21 transform function from the matlab and references interpretation
      // frame->data[0][y * frame->linesize[0] + x] = x + y;  // testing for the filter
to work
    }
  }

  return ff_filter_frame(inlink->dst->outputs[0], frame);
}

static const AVFilterPad pu21_inputs[] = {
    {
        .name = "default",
        .type = AVMEDIA_TYPE_VIDEO,
        .filter_frame = filter_frame,
    }
};

static const AVFilterPad pu21_outputs[] = {
    {
        .name = "default",
        .type = AVMEDIA_TYPE_VIDEO,
    }
};

const AVFilter ff_vf_pu21 = {
    .name = "pu21",
    .description = NULL_IF_CONFIG_SMALL("Convert HDR linear values to PU values using 
PU21 transform"),
    .priv_size = sizeof(PU21Context),
    .init = pu21_init,
    FILTER_INPUTS(pu21_inputs),
    FILTER_OUTPUTS(pu21_outputs),
    .priv_class = &pu21_class,
    .flags = AVFILTER_FLAG_SUPPORT_TIMELINE_GENERIC,
};
```

Addition of tests for the new filter

The FFmpeg project has an integrated testing pipeline named FATE (FFmpeg Automated Testing Environment). It tests regression to verify that patches work similarly in various environments and systems. Adding FATE tests would be an integral part of the project, ensuring that the transformation works equally on all devices.

Fortnightly blogs on developmental advances and milestones

I maintain my personal writing space in rajivharlalka.in which is publicly available and with RSS support for easy updates. I want to leverage this space to talk about my learning and understanding on the way with at least one blog each fortnight. I also infrequently maintain personal notes (though public), which are hand-crafted utilities for myself to fasten things up at notes.rajivharlalka.in. This would also be a place where my learning would be frequently updated.

. Brief Timeline

Pre-GSoC Period

Till 15 April:

I plan to utilize this period mostly for brushing up my C-skills and gain an understanding how the different audio and video filters are implemented. This would help when the actual coding period starts and reduce a lot of effort that time. Reading through the mailing list with the latest patches is also something I plan on to keep myself updated with the latest development. Understanding more on the background of the problem statement through research papers would also be executed during this period as it would help in making me have a clearer picture of the requirements and the deliverables.

Community Bonding Period [3 Weeks]

Being active on the Mailing List for the last few weeks have given me quite an understanding of how the community works and the different aspects to the project as a whole, though I still believe there is a lot for me to see and understand. Understanding more on the utilization aspect of the tool, it’s helper tools are things which I plan to gain a bit of experience on in this period. Would also increase my visibility on the mailing list and the IRC channel loop myself in with the community members. Would also establish one-on-one meet with the mentor on a suitable time for both ends.

At the end of this period, I aim to:

1. Have a clearer set of understanding about the project.
2. Better grasp on necessary skill sets.

Coding Period 1 [27 May - 8 July]

By this time I would have a better sense of which direction I need to work on, I would start with the implementation of the filter. With having the first draft of the entire filter ready within the next three weeks, I would start on optimizations and community feedback in areas of improvement and standards I might have missed out.

Phase - 1 Evaluations

Expected Outcomes at this stage:

- Filter for PU21 transform on the mailing list with at least some on-going discussions.

Coding Period 2 [12 July - 26 August]

The second leg of the program would be finally invested in adding tests, documentation and extensive blogs and ways to explain the utilization and benefit of the transformation of HDR images in quality testing.Any remaining discussions over the mailing list would also be wrapped over in this period itself.

I would utilize this time to complete any leftover nit picking. I would also use this time as a buffer period in case of any lag in the schedule.

In case of no delays, I would use this time for some post-GSoC work mentioned below.

Phase - 2 Evaluations

Post GSoC Plans:

I’m **not** applying for GSoC under any other organization this year since I would like to help in the development of the FFmpeg which I have been utilizing directly/ indirectly for quite a long time and thus spending my summer holidays meaningfully.

I would like to extend my work with helping in writing tests and documentations for a number of parts which I have felt even right now missing. For the parts of documentation, there were a number of things I had to look at, which if aggregated might help future contributors.

At this point I also feel the entry barrier to the community and project is relatively higher considering a number of factors, discussions over IRC/ML rather than latest chat applications, codebase in C/ASM which might be daunting for new people initially. I would try to help newcomers by writing blogs on such topics and workflows which are helping me right now.

Additional Information about the Timeline

- The timeline mentioned above is subject to change and is only an approximate outline of my project work. I will stick to or exceed this schedule and create a more detailed schedule during the pre-GSoC and community bonding phase.
- Fortnightly blogs will be maintained at https://rajivharlalka.in and will include highlights of the development process and methods used to overcome hurdles.

References:

Links have been added at appropriate locations throughout the text, directing to multiple relevant references. Below are some other resources relevant to the context of the proposal.

- Mantiuk, Rafal & Azimi, Maryam. (2021). PU21: A novel perceptually uniform encoding for adapting existing quality metrics for HDR. 1-5. 10.1109/PCS50896.2021.9477471.
- Mikhailiuk, A., & Mantiuk, R.K. (2021). The effect of display brightness and viewing distance: a dataset for visually lossless image compression. *Electronic Imaging*.
- Rafal Mantiuk, Grzegorz Krawczyk, Karol Myszkowski, and Hans-Peter Seidel. 2004. Perception-motivated high dynamic range video encoding. ACM Trans. Graph. 23, 3 (August 2004), 733–741. https://doi.org/10.1145/1015706.1015794
- Daly, Scott J.. “Visible differences predictor: an algorithm for the assessment of image fidelity.” *Electronic imaging* (1992).